Europäisches Patentamt
European Patent Office
Office européen des brevets

Bescheinigung Certificate Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. Patent application No. Demande de brevet n°

02291795.9

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH
RULE 17.1(a) OR (b)

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

R C van Dijk

Europäisches Patentamt | European Patent Office | Office européen des brevets

Anmeldung Nr:
Application no.: 02291795.9
Demande no:

Anmeldetag:
Date of filing: 17.07.02
Date de dépôt:

Anmelder/Applicant(s)/Demandeur(s):

SCHLUMBERGER Systèmes
50, avenue Jean Jaurès
92120 Montrouge
FRANCE

Bezeichnung der Erfindung/Title of the invention/Titre de l'invention:
(Falls die Bezeichnung der Erfindung nicht angegeben ist, siehe Beschreibung.
If no title is shown please refer to the description.
Si aucun titre n'est indiqué se referer à la description.)

USB device

In Anspruch genommene Priorität(en) / Priority(ies) claimed /Priorité(s) revendiquée(s)
Staat/Tag/Aktenzeichen/State/Date/File no./Pays/Date/Numéro de dépôt:

Internationale Patentklassifikation/International Patent Classification/
Classification internationale des brevets:

G06F13/00

Am Anmeldetag benannte Vertragstaaten/Contracting states designated at date of filing/Etats contractants désignées lors du dépôt:

AT BE BG CH CY CZ DE DK EE ES FI FR GB GR IE IT LI LU MC NL PT SE SK TR

USB device

The USB protocol presents a hot plug and play capabiliity. It means that when a USB device is plug onto a USB host, an "enumeration phase" allows to the USB host to chose one or more driver(s) to load or to install from the local host (if the device is already installed or is a standard device) or from an external source (floppy disk, CD-ROM, network ...) if the device is not installed.

A USB device is organized with several levels, each level being represented by a USB descriptor :

- the *device descriptor* describes the overall device. It is associated to one or more configuration descriptors,
- the *configuration descriptor* describes some electrical characteristics of the device, or a part of the device. It is associated to one or more interface descriptors,
- the *interface descriptor* describes a particular application of the device. An interface can contain one or more alternate settings. An interface descriptor is associated to zero or more endpoint descriptors,
- the *endpoint descriptor* describes a communication channel used by the application defined by the interface descriptor.

The USB device is seen by the USB host as a services provider. This(these) service(s) can be offered at the device level (standard USB device), or at the interface level (composite USB device).

The number of drivers loaded depends on the number of different services present in the device. For example, if a device is at the same time a scanner and a printer, from a USB point of view, it will have to present two interfaces during the enumeration phase. In that case, two drivers, one associated to the scanner interface, and the other one associated to the printer interface, will be loaded after the enumeration phase. A third driver associated to the device itself can also be loaded (composite device driver relative to the Manufacturer and Product ID of the device).

On the other hand, some standard devices are defined thrue USB Class Devices Specification. Any USB host should implement the drivers relative to the USB Class devices. The other drivers (Vendor Specific drivers) must be funished with the device in order to run it corroctly.
It supposes that any Vendor Specific, or not standardized devices should be accompanied with the corresponding driver(s) on a disk, or that the driver(s) should be accessible from the network.

Moreover, some devices may require some particular applications associated to there functionnalities. These application should follow the same scheme as the driver(s) concerning the distribution.

Selfdriven USB device :

Following the scheme described above, it is possible to imagine an "autodriven" device. Indeed, the USB-IF has developped a Mass Storage Device Class with a standard driver that can be used for example to drive the disk readers.

A Mass Storage service can be implemented inside any USB device in order to make the device containing its own driver(s) and/or its own application(s).
In that way, the device can be used in any USB host, even if the driver(s) for the device are not installed nor available, since the driver(s) is(are) available in the device itself.

The device could have the following configuration : SEE FIGURE 1

In the previous case, the device contains two standard interfaces, and a number N of Vendor Specific interfaces. When the device is plugged onto a USB port, it presents only the standard interfaces to the USB host.
From the user point of view, the device is seen, among other things, as a disk. It is then possible to install a driver or an application for the device from the device.

After the driver and/or application installation, the device can be detached, or can detach itself from the bus, and present all its interfaces once it is replugged.

A second scenario can also be envisaged :
- The device presents all its interfaces to the host during the enumeration phase,
- Some drivers are not available, so the corresponding interfaces will be marked as not correctly installed,
- The user can update the drivers from the device after plugging phase completion.

To be completely efficient, it is an advantage to define an appropriate protocol in a standard device class, in order to have an *actors identification*. During this phase managed by the standard class device associated to the device, the host can indicate to the device its nature, in order the device to chose the drivers and/or application that must be made available for the host to install.
This phase could be completed using a USB request. The information transported by this request could be the computer type, and the operating system used.

Example :

The device is a Smart Card embeding three different services :

- Keys and rights management (APDU command transport) as standard class service [S1],
- Mass storage as standard class service [S2],
- Internet Service Provider login application (DRM) as vendor specific service [S3].

When the Smart Card is plugged onto a USB port, it presents only the interfaces [S1] and [S2].
During the enumeration phase, the USB host loads the driver associated to [S1] and [S2], and the driver of [S1] starts the negotiation phase.

A USB request allows the Host to indicate which type of computer is used, and which OS is installed. In that way, the smart card can chose a specific driver linked to the host environment, and present the appropriate driver and application necessary to run [S3].
Once the driver is installed in the host environment, the smart card detaches itself from the bus, and re-attaches itself. During the enumeration phase that follows, the smart card presents the services [S1], [S2] and [S3]. The user can then use the application directly from the device to access to [S3].

In that way, it is possible to imagine that any Internet Service Provider can define its own login application. This application can not be modified by a hacker (since it is stored on the card), and is more difficult to hack, since it is not a standard implementation.

Remarks :

- This memo is applicable to any USB device,
- The memo describes a USB system that can be easily personalized by a USB device itself.

Figure 2 illustrates an example wherein the smartcard (USB device) may also comprise a loudspeaker service, a microphone service and a decryption service. Once the smartcard (USB device) is plugged into a computer (USB host) the loudspeaker service will be activated by the computer. The computer will first consider the smartcard (USB device) as a loudspeaker and will send an encrypted music file (1). The loudspeaker service will receive the music file and send it (2) to the decryption service for decrypting the music file. Then the decrypted music file is sent (2) to the microphone service so that the computer (USB host) believes now (3) that the smartcard (USB device) is a microphone wherein someone is speaking. The computer will then send the decrypted music file to the real loudspeaker (4).

It should be clear that the invention is not limited to devices communicating using the USB protocol. Other protocol like, for example, firewire based protocol may be used.

The description hereinbefore illustrates the following features:

A system comprising a main device and an auxiliary device arranged to co-operate with each other, the auxiliary device being arranged to effect a core functionality, the auxiliary device comprising descriptors, characterised in that the auxiliary device comprises at least one descriptor that defines a functionality that is different from the core functionality.

According to another aspect of the invention, the main device is, for example, a USB host and the auxiliary device is, for example, a USB device.

According to another aspect of the invention, the functionality that is different from the core functionality is, for example, a mass storage functionality

According to another aspect of the invention, the auxiliary device, when it is coupled to the main device, initially presents the descriptor that defines a functionality that is different from the core functionality.

According to another aspect of the invention, the USB device comprises a driver for the USB host to be installed by simulating that the USB device is a mass storage.

According to another aspect of the invention, the USB device is a smartcard.

CLAIMS

1. A system comprising a main device and an auxiliary device arranged to co-operate with each other, the auxiliary device being arranged to effect a core functionality, the auxiliary device comprising descriptors, characterised in that the auxiliary device comprises at least one descriptor that defines a functionality that is different from the core functionality.
2. The system according to claim 1, wherein the main device is a USB host and the auxiliary device is a USB device.
3. The system according to claim 2, wherein the functionality that is different from the core functionality is a mass storage functionality.
4. The system according to claim 1, wherein the auxiliary device, when it is coupled to the main device, initially presents the descriptor, that defines a functionality that is different from the core functionality.
5. The system according to claim 2, wherein the USB device comprises a driver for the USB host to be installed by simulating that the USB device is a mass storage.
6. The system according to claim 2, wherein the USB device is a smartcard.

ABSTRACT

A system comprises a main device and an auxiliary device arranged to co-operate with each other. The auxiliary device is arranged to effect a core functionality. The auxiliary device comprises descriptors. The system is characterised in that the auxiliary device comprises at least one descriptor that defines a functionality that is different from the core functionality.

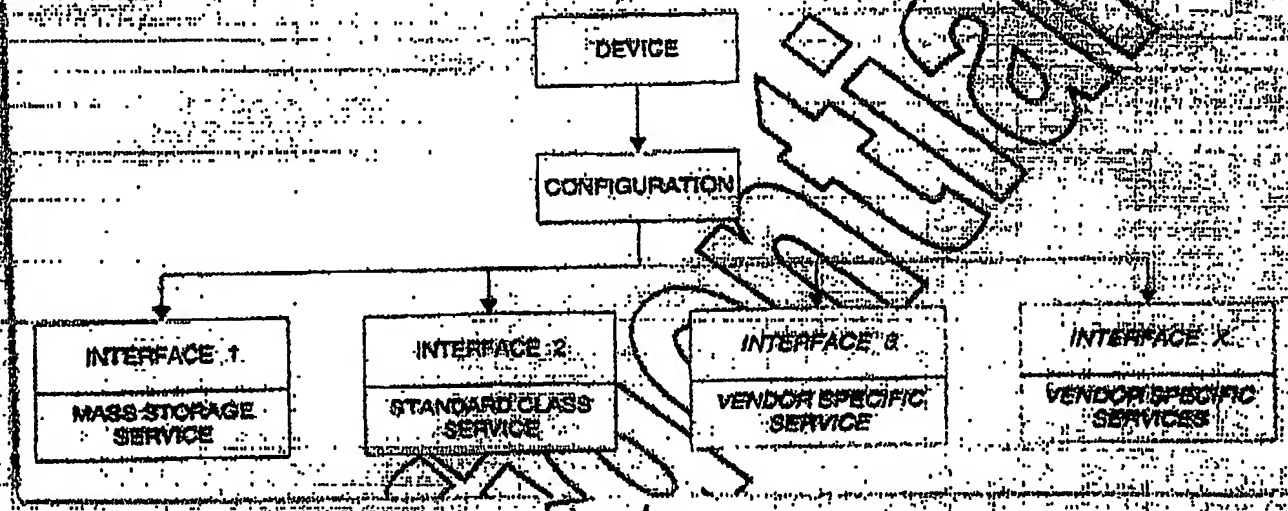
DEVICE
CONFIGURATION
INTERFACE 1
MASS STORAGE SERVICE
INTERFACE 2
STANDARD CLASS SERVICE
INTERFACE &
VENDOR SPECIFIC SERVICE
INTERFACE X
VENDOR SPECIFIC SERVICES

Fig 1

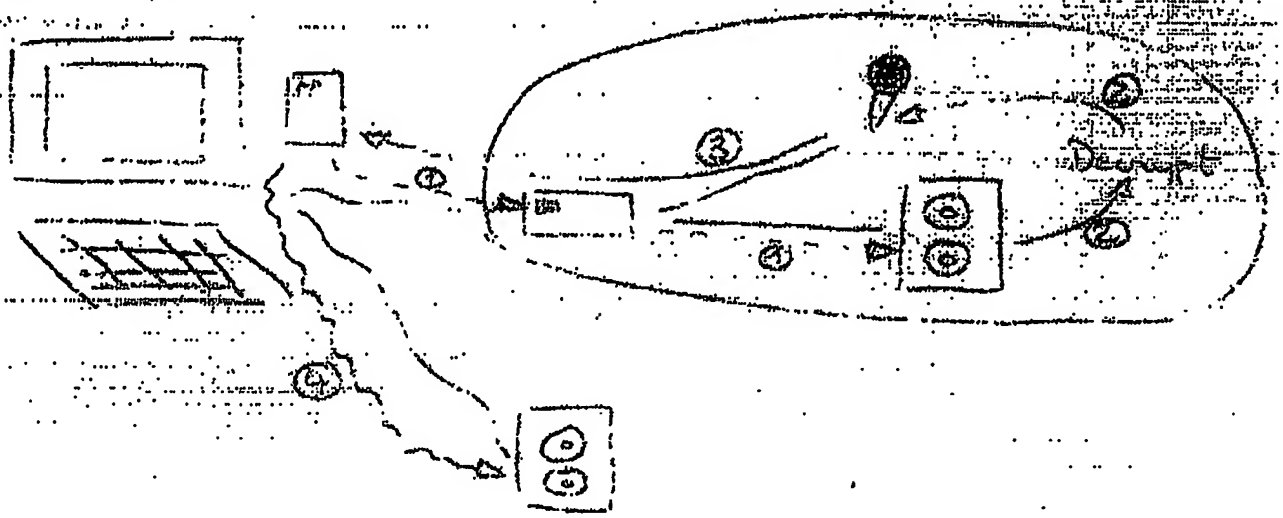

Fig 2